



SCHOOL OF COMPUTER AND INFORMATION SCIENCES
FACULTY OF DESIGN AND CREATIVE TECHNOLOGIES
BACHELOR OF COMPUTER AND INFORMATION SCIENCES

General Synopsis

405704

Programming 2

1	PAPER INFORMATION	2
1.1	Prescriptor	2
1.2	Content	2
1.3	Learning Outcomes	3
1.4	Learning and Teaching Strategies	3
1.5	Duration & Student Workload	4
1.6	Teaching Team	5
1.7	Class Times and Places	5
1.8	Communication via Email	5
1.9	AUTOnline Announcements	5
2	OVERVIEW OF THE SEMESTER	5
2.1	Course Calendar Semester 1 2006	6
3	ASSESSMENT	7
3.1	Assessment Items Summary	7
3.2	Course Requirements	8
3.3	Hand-Back	8
3.4	Assessment Regulations	8
3.5	Academic Honesty and Integrity	8
4	READINGS	9

1 PAPER INFORMATION

Paper Title:	Programming 2
Paper Code:	405704
Prerequisites:	405701, Programming 1 405703, IT Hardware & Software Infrastructure
Co-requisites:	None
Level:	5
Points:	15

1.1 Prescriptor

To further develop the programming skills of the student by introducing advanced Object-Oriented programming concepts and simple graphical user interfaces.

1.2 Content

- Inheritance and polymorphism
- Abstract classes
- Interfaces
- A professional IDE
- Java Swing and GUIs
- Reading and writing to files
- Data structures
- Algorithm design
- Software design
- Simulations

1.3 Learning Outcomes

By the end of programming 2 students will be able to:

- explain inheritance and polymorphism
- explain the concept and uses of abstract classes
- explain the concept and uses of interfaces
- explain and apply recursion in an appropriate context
- use Tree data structures
- explain Tree data structures
- describe advanced algorithms (eg: sorting or searching)
- use inheritance in the correct programming context
- use abstract classes in the correct programming context
- use interfaces in the correct programming context
- employ exception handling and create exception classes
- apply the principles of serialisation
- interpret software requirements
- develop software with a modular design
- demonstrate the skills used in effective software testing
- explain the processes involved in software testing
- use automated software testing tools
- develop a simple graphical user interface
- evaluate and use a variety of data structures
- design, evaluate and implement efficient algorithms
- program from scratch a 'large-scale' piece of software
- design a 'large-scale' piece of quality software from a provided specification
- implement a search algorithm
- explain and determine the order of magnitude of a searching algorithm
- use file input and output in Java to store data including serializable objects
- name and explain the role of 3 design patterns
- explain the advantages and motivations for using design patterns

1.4 Learning and Teaching Strategies

- **Lectures** – lecturers will introduce and emphasise key concepts for each of the topics listed in the syllabus. Some time may be used to design solutions to exercises.
- **Tutorials (lab sessions)** – students will work to generate solutions to problems, test and discuss their solutions. They will become familiar with one programming language.
- **Review and research** – students will review and research key concepts introduced by lecturers, to broaden base knowledge. Lecture notes are provided (see AUTOnline) but should be expanded by students so that they develop a full understanding of each topic. Mere memorisation of the notes is not adequate preparation for the assessments.
- **AUTOnline** – All lecture notes, exercises and assignment details for the paper are available through AUTOnline. Students are also encouraged to use the Discussion Forums to seek help and to share their knowledge and experiences.

Class Attendance: All classes should be attended to ensure your success in this paper. Lab sessions are particularly important as it is here that students really learn to program. The Programme Leader may withdraw a student's enrolment in a paper if the student is absent for a consecutive period that constitutes more than 20% of the scheduled hours for that paper.

1.5 Duration & Student Workload

<i>Item</i>	<i>Weeks</i>	<i>Hours</i>
Teaching	13 weeks * 2 hrs per week	26
Revision & Examination	2 weeks * 2 hrs	4
Tutorials	13 weeks * 2 hrs per week	26
Independent Study		94
Total		150
Independent Study:	6-8 hours weekly through-out the semester. In total 150 hours of student study time is expected.	

1.6 Teaching Team

This will vary from semester to semester.

1.7 Class Times and Places

You can access your personal timetable using your Arion student login name and password from <https://arion.aut.ac.nz/>. Once logged into the Arion website, you can find the timetable for this paper on:

<https://arion.aut.ac.nz/ArionMain/CourseInfo/Information/Qualifications/Details/PaperDetails.aspx?actiontype=1&id=32585>

1.8 Communication via Email

It is our policy that all email communication with students is conducted using their AUT email address. Please ensure that your AUT email address is regularly checked as important information might be sent to you during the semester. It is students' responsibility to check their email box and/or arrange for forwarding AUT email correspondence to another [private] address.

AUTOnline contains information on forwarding AUT email to an external email address.

1.9 AUTOnline Announcements

It is the policy of this paper to make general announcements to students through AUTOnline. We assume that once an announcement is made, students will read it within a reasonably short time frame.

2 OVERVIEW OF THE SEMESTER

The course calendar that follows describes the topics for each week. The calendar as presented is subject to change. Notification will be given (via AUTOnline) if this occurs.

2.1 Course Calendar Semester 2 2006

WEEK	CONTENT	ASSESSMENTS	BOOK CHAPTERS
1	Summary of Programming 1. Inheritance.		8
2	Subclasses and sub types. Polymorphism.		8
3	Simulations. Abstract classes.		9
4	Interfaces. Test preparation. Designing applications CRC cards.	Test 1	10
5	Designing applications.	Assignment stage design	13
6	User interfaces. Events and event handling.		11
7	Beyond BlueJ - another IDE	Assignment implementation stage	11
8	Handling errors, defensive programming. Design by contract.		12
9	File IO. Object serialisation.	Test 2	12
10	Designing applications - design patterns. Case study.	Assignment review	13-14
11	Case study.		14
12	Data structures. Recursion.	Assignment due	
13	Algorithm design. Order of magnitude.	Test 3	
14	NO CLASSES : Revision week		
15	NO CLASSES : Exam week		

3 ASSESSMENT

3.1 Assessment Items Summary

Item	Weight %	Date Given	Date Due
Class tests	30	Weeks 4, 9 and 13	
Assignment	30	Week 5	Week 12
Theory Examination	40		Week 14/15

3.1.1 Late Acceptance Policy (when no extension has been granted)

Assignments will be accepted up to 2 days late with a 5% per day late penalty..

3.1.2 Pass Mark

To pass the paper, the student needs to obtain at least a minimum pass in each assessment item and to obtain at least 50% overall.

3.1.3 Written Assessment Presentation Guidelines

In Programming 2, most written assessments will consist of printed code, diagrams and tables of results. The standards required for these are given on AUTOonline. Where written descriptive work is required, please note the following general points:

- Supply a correctly written reference list and a bibliography where you have referenced external sources.
- Use a black 12 point font for the main body of work.
- Number the pages.
- Include your name and ID number on each page as a header or a footer
- Include a table of contents page and appropriately numbered the sections.

Please also note the following:

- Use a header at the start of a section to make it clear which assessment item is being answered.

- Check assignments for grammar and spelling errors before submitting them.
- In addition to other required deliverables, provide an electronic copy of your assignment for the purposes of academic quality control.

3.2 Course Requirements

Students who do not meet the course requirement will not be eligible to complete the course. To meet the requirement for this paper, follow the instructions and due dates as outlined in Assessment Summary.

3.3 Hand-Back

All assessment items that have not previously been returned will be handed back at the end of semester. Hand-back date, time and procedure will be advised during the semester. Hand-back is the only opportunity students have to check exam marks and request reconsideration. Attendance is thus highly recommended.

3.3.1 Uncollected Assessment Work

Paper coordinators will hold all uncollected scripts for 3 weeks after hand-back when they will be destroyed.

3.4 Assessment Regulations

This paper operates according to the School of Computer and Information Sciences (SCIS), Faculty of Design and Creative Technologies regulations and the Auckland University of Technology General Academic Statutes.

You are advised to become familiar with the contents of the SCIS regulations available via the School website (<http://www.aut.ac.nz/cis>) and the AUT Academic Calendar (<http://www.aut.ac.nz/students/>).

3.5 Academic Honesty and Integrity

The University regards most seriously any acts of dishonesty relating to assessments. Any such acts may result in a mark of zero for the assessment and may lead to disciplinary action.

Examples of dishonesty include:

- Plagiarism.
- Unauthorised collaboration.
- Examination misconduct.
- Theft of other student's work.
- Use of previously submitted assignment work.

3.5.1 Plagiarism

Plagiarism means borrowing from the work of another without indicating by referencing (and by quotation marks where exact phrases are borrowed) that the ideas expressed are not one's own. Students may use the ideas and information of other authors, but this use must be acknowledged. It is not acceptable to submit an assignment which is simply paraphrasing extracts from other authors: the work submitted must include a recognisable intellectual contribution by the student.

3.5.2 Unauthorised Collaboration

Unauthorised collaboration means joint effort between students or students and others, in preparing material submitted for assessment, except where this has been approved by the course programme. Students are encouraged to discuss matters covered in classes, but when writing an assignment, report, essay, or other piece of assessed work, the recording and treatment of the data and the expression of ideas and argument must be the student's own work.

It is also considered to be inappropriate to give code to another student where this code relates to an assessment. If two pieces of similar work are submitted for assessment, both students will be liable for disciplinary action. Assisting fellow students with their assessment work must be limited to general discussions.

4 READINGS

4.1 Required texts

David J. Barnes and Michael Kolling: *Objects First with Java A PRACTICAL INTRODUCTION USING BlueJ*; Pearson:Prentice Hall, Second Edition, 2005; ISBN 0131243933 9

(Or Third edition ISBN-13: 978-0-13197-629-0)

This text is **essential** and will be required for pre-lecture readings and exercises in this book form the basis of the laboratory classes. In addition the disk that ships with this book provides you with sample projects and the BlueJ development environment.

4.2 Recommended texts

The following text will be of some help, although it is a general book and was not written specifically for this paper.

Main, Michael *Data Structures And Other Objects Using Java*, Addison Wesley, ISBN 0-201-74093-1

4.4 AUT Online

Course information and materials are available over the internet from <http://www.autonline.ac.nz>